

09/520378

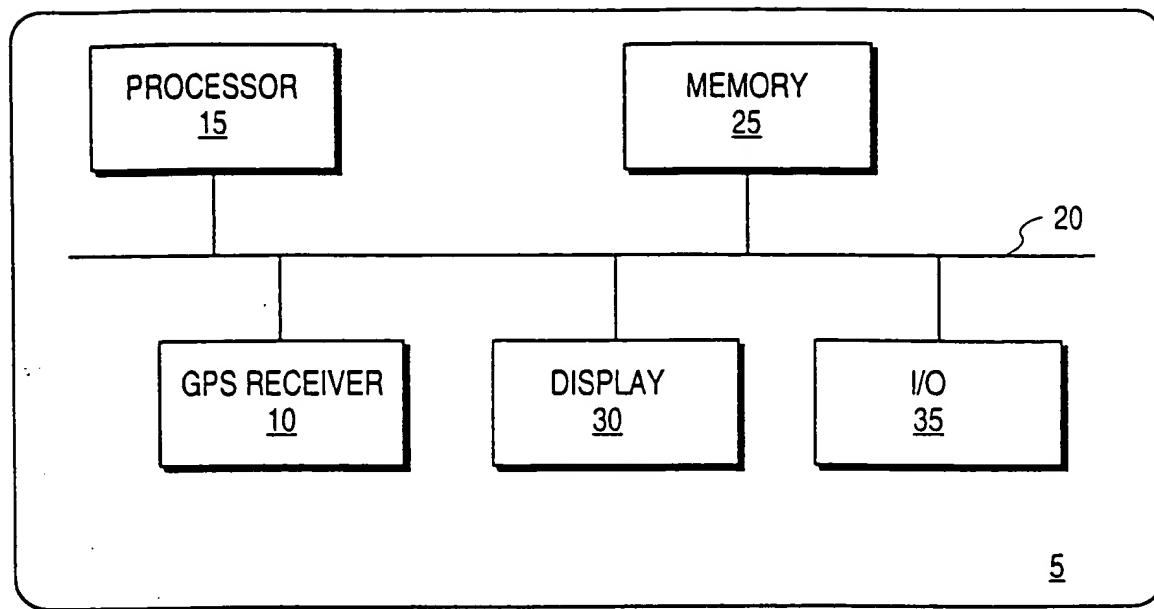


FIG. 1

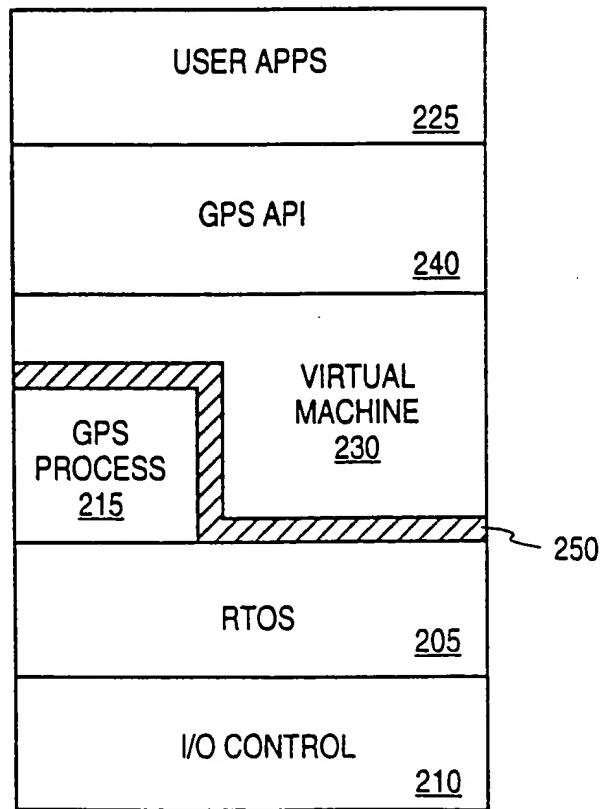


FIG. 2

CLASS GPS.ROUTEPOINT	
METHODS	
-getLat	public double getLat ()
-getLon	public double getLon ()
-getTime	public int getTime ()
-getRadius	public int getRadius ()
-getEarlyThreshold	public int getEarlyThreshold ()
-getLateThreshold	public int getLateThreshold ()
-getDistance	public double getDistance ()
-getIndex	public int getIndex ()
-setTime	
-setRadius	public void setRadius (int radius)
-setEarlyThreshold	public void setEarlyThreshold (int early)
-setLateThreshold	public void setLateThreshold (int late)
-setDistance	public void setDistance (double distance)

FIG. 3b

CLASS GPS.ROUTEPOINT	
METHODS	
-setIndex	public void setIndex (int index)
-toString	public String toString ( ) <b>Overrides:</b> toString in class Object

CLASS GPS.GPSTIME	
VARIABLES	
-SECS_PER_WEEK	public static final int SECS_PER_WEEK
-SECS_PER_DAY	public static final int SECS_PER_DAY
-SECS_PER_HOUR	public static final int SECS_PER_HOUR
-SECS_PER_MINUTE	public static final int SECS_PER_MINUTE
-MINS_PER_HOUR	public static final int MINS_PER_HOUR
-HOURS_PER_DAY	public static final int HOURS_PER_DAY
-DAYS_PER_WEEK	public static final int DAYS_PER_WEEK

FIG. 4a

CLASS GPSS.GPSTIME	
CONSTRUCTORS	
-GPSTime	public GPSTime () Constructs a GPSTime object with the current date and time
-GPSTime	public GPSTime (int yyyy, int m, int d) Constructs a specific GPSTime given only the date <b>Parameters:</b> yyyy - year (full year, e.g., 1996, not starting from 1900) m - month (1-12) d - day (1-31) <b>Throws:</b> IllegalArgumentException if yyyy/m/d h:min:ss not a valid date/time
-GPSTime	public GPSTime (int yyyy, int m, int d, int h, int min, float s) Constructs a specific GPSTime given a date & time <b>Parameters:</b> yyyy - year (full year, e.g., 1996, not starting from 1900) h - hour (range 0-23) min - minute (range 0-59) s - second (range 0-59.999...) <b>Throws:</b> IllegalArgumentException if yyyy/m/d h:min:ss not a valid date/time

FIG. 4b

FIG. 4c

CLASS GPS.GPSTIME	
CONSTRUCTORS, cont.	
GPSTime	<p>public GPSTime (short week_tag, float time_tag)</p> <p>Constructs a specific GPSTime given the GPS week/second tags. This method corrects for UTC leap seconds and performs GPS week rollover checking according to the current rollover threshold currently in effect</p> <p><b>Parameters:</b></p> <p>week_tag - GPS week number (range 0 to 1023) time_tag - Seconds into the GPS week (not adjusted for UTC)</p>

CLASS GPS.GPSTIME	
METHODS	
-advanceDay	<p>public void advanceDay (int n)</p> <p>Advance by n days. For example. d.advanceDay(30) adds thirty days to d</p> <p><b>Parameters:</b></p> <p>n - the number of days by which to change this (n can be &lt; 0)</p>
-advanceSecond	<p>public void advanceSecond (float n)</p> <p>Advance the time by n 'seconds'. For example. d.advanceSecond(30) adds thirty seconds to d</p> <p><b>Parameters:</b></p> <p>n - the number of seconds by which to change this day (can be &lt; 0)</p>
-getSecond	<p>public float getSecond ()</p> <p>Gets the second of the minute</p> <p><b>Returns:</b></p> <p>the second of the minute (range 0 to 59.999...)</p>
-getMinute	<p>public int getMinute ()</p> <p>Gets the minute of the hour</p> <p><b>Returns:</b></p> <p>the minute of the hour (range 0 to 59)</p>
-getHour	<p>public int getHour ()</p> <p>Gets the hour of the day</p> <p><b>Returns:</b></p> <p>the hour of the day (range 0 to 23)</p>

CLASS GPS.GPSTIME		
METHODS cont.		
-getDay	public int getDay ()	Gets the day of the month <b>Returns:</b> the day of the month (range 0 to 31, month dependent)
-getMonth	public int getMonth ()	Gets the month <b>Returns:</b> the month (range 1 to 12)
-getYear	public int getYear ()	Gets the year <b>Returns:</b> the year (counting from 0, <i>not</i> 1900)
-weekday	public int weekday ()	Gets the weekday <b>Returns:</b> the weekday (0 = Sunday, 1 = Monday, ..., 6 = Saturday)
-daysBetween	public int daysBetween (GPSTime b)	The number of days between this and GPSTime parameter <b>Parameters:</b> b - any GPSTime <b>Returns:</b> the number of days between this and GPSTime parameter and b (> 0 if this day comes after b)

FIG. 4e

CLASS GPS.GPSTIME		
METHODS cont.		
-secsBetween public double secsBetween (GPSTime b)	The number of seconds between this and GPSTime parameter <b>Parameters:</b> b - any GPSTime <b>Returns:</b> the number of seconds between this and GPSTime parameter and b (> 0 if this comes after b)	
-getWeek_tag public short getWeek_tag ()	Get the GPS week_tag <b>Returns:</b> the GPSweek_tag value (aliased to lie from 0 - 1023)	
-getTime_tag public float getTime_tag ()	Get the GPS time_tag <b>Returns:</b> the GPSTime_tag value (offset from UTC by GPS leap seconds)	
-convertGPSTimetag public void convertGPSTimetag (short week_tag, float time_tag)	Set this GPSTime to the GPS week/seconds tags. This method corrects for UTC leap seconds and performs GPS week rollover according to the current rollover threshold currently in effect <b>Parameters:</b> week_tag - GPS week number (range 0 to 1023) time_tag - Seconds into the GPS week (not adjusted for UTC)	

CLASS GPS.GPSTIME			
METHODS cont.			
- <b>toString</b>	public String <b>toString</b> ( )	A string representation of the day <b>Returns:</b> a string representation of the GPS date and time <b>Overrides:</b> <u>toString</u> in class Object	
- <b>DurationString</b>	public static String <b>DurationString</b> (int dt)	A string representation of a duration in seconds <b>Parameters:</b> dt - Delta time in seconds <b>Returns:</b> a string representation of the delta seconds parameter	
- <b>toCalendar</b>	public Calendar <b>toCalendar</b> ( )	Convert to Java Calendar object using the default Time zone and locale GPS seconds round to the nearest integer second	
- <b>clone</b>	public Object <b>clone</b> ( )	Makes a bitwise copy of a GPSTime object <b>Returns:</b> a bitwise copy of a GPSTime object <b>Overrides:</b> <u>clone</u> in class Object	
- <b>main</b>	public static void <b>main</b> (String args [ ])		

CLASS GPS.GPSFIX		
METHODS		
-clone	public Object clone ()	Makes a bitwise copy of a GpsFix object <b>Returns:</b> a bitwise copy of a SimFix object TBD: sub-objects must also support cloning and be explicitly cloned here. <b>Overrides:</b> clone in class Object.
-getDGPSSflag	public boolean getDGPSSflag ()	Get the Differential GPS status of the current fix. A TRUE value may be either 2D or 3D.
-GetLatitude	public double GetLatitude ()	Get the latitude in degrees referenced to WGS-84 Positive values indicate northern hemisphere. Negative values indicate southern hemisphere.
-GetLongitude	public double GetLongitude ()	Get the longitude in degrees referenced WGS-84 Negative values indicate western hemisphere. Positive values indicate eastern hemisphere.
-GetAltitudeMSL	public double GetAltitudeMSL ()	Get the altitude in meters above the geoid (mean sea-level)
-getAltitudeWGS84	public double getAltitudeWGS84 ()	Get the altitude in meters above the WGS-84 ellipsoid.
-getTimeTag	public float getTimeTag ()	Get the GPS time tag as seconds within the GPS week.

**FIG. 5a**

## CLASS GPS.GPSFIX

### METHODS cont.

-getWeekTag	public short getWeekTag ()	Get the GPS week tag (0-1023) from the GPS epoch. This epoch is nominally Jan 6, 1980, but can be adjusted accordingly within the GPSTime class.
-getTimeOfFix	public GPSTime getTimeOfFix ()	Return the UTC (leap-second corrected) time of current fix.
-AgeOfFix	public double AgeOfFix ()	Get the age of the current fix in seconds as compared to (GPS-corrected) system time.
-TimeSincePreviousFix	public float TimeSincePreviousFix (GpsFix prevfix)	Return the number of seconds between this fix and the specified (prior) fix.
-GetSpeed	public float GetSpeed ()	Return the horizontal speed in meters per second.
-GetHeading	public float GetHeading ()	Return the current "course" in degrees clockwise from the true north.
-GetVspeed	public float GetVspeed ()	Return the vertical speed in meters per second.
-equals	public boolean equals (GpsFix f)	Return true if fixes are equal.
-print	public void print (String s)	
-print	public void print ()	

**FIG. 5b**